AFHRL-TP-88-57

AIR FORCE HUMAN RESOURCES LABORATORY

INFLITE: AN INTELLIGENT INSTRUMENT FLIGHT TRAINER WITH COMPUTER-GENERATED SPEECH

J. Wesley Regian

TRAINING SYSTEMS DIVISION
Brooks Air Force Base, Texas 78235-5601

Margaret M. Dennis

Pentastar, Incorporated
200 West 7th Street, Suite 1515
Forth Worth, Texas 76102

Valerie J. Shute

MANPOWER AND PERSONNEL DIVISION
Brooks Air Force Base, Texas 78235-5601

March 1989
Interim Technical Paper for Period May - November 1988

AIR FORCE SYSTEMS COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5601

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this paper, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This paper has been reviewed and is approved for publication.

HENDRICK W. RUCK, Technical Advisor
Training Systems Division

RODGER D. BALLENTINE, Lt Col, USAF
Chief, Training Systems Division

AFHRL Technical Paper 88-57 March 1989

INFLITE: AN INTELLIGENT INSTRUMENT FLIGHT TRAINER WITH COMPUTER-GENERATED SPEECH

J. Wesley Regian

TRAINING SYSTEMS DIVISION
Brooks Air Force Base, Texas 78235-5601

Margaret M. Dennis

Pentastar, Incorporated
200 West 7th Street, Suite 1515
Forth Worth, Texas 76102

Valerie J. Shute

MANPOWER AND PERSONNEL DIVISION
Brooks Air Force Base, Texas 78235-5601

Reviewed by

J. Wesley Regian
Intelligent Systems Branch

Submitted for Publication by

Hugh L. Burns, Lt Col, USAF
Chief, Intelligent Systems Branch

Paper presented at the Technology and Innovations in Training and Education Conference, Atlanta, Georgia, 6-10 March 1989.

SUMMARY

This paper describes an Intelligent Tutoring System (ITS) developed at the Air Force Human Resources Laboratory (AFHRL) that teaches cognitive skills associated with performing an instrument landing in a fighter aircraft. The ITS is presented not as an actual training device for instrument flight, but as an example of the application of artificial intelligence (AI) to training in a class of task domains which we refer to as high-performance tasks (Regian & Shute, 1988). In high-performance tasks, there is a greater requirement for speeded, reliable, and automatic task performance than is found in the typical knowledge-rich ITS domains (e.g., medical diagnosis, electronic troubleshooting). An Instrument Flight Trainer (INFLITE) was developed in the Training Systems Division at AFHRL to test the concept of using AI to train high-performance tasks. The prototype system trains students to land a simulated aircraft (F-16) using instruments only. During the process, an intelligent coach monitors the student and provides guidance just as an instructor pilot might guide a student pilot. This guidance is presented verbally, using a speech synthesis system to simulate human speech. The student also receives instructions from a simulated Air Traffic Controller, again via speech synthesis but with a different voice from that of the coach. During early training sessions, the coach has the ability to freeze the simulation to give guidance. The coach also debriefs the student after each training session and prebriefs the student before each training session. During the post-flight debriefs, the coach reviews the student's performance in comparison to performance on earlier flights and highlights specific areas to be worked on in future flights. During the preflight briefs, the coach reminds the student of problem areas that were identified in earlier flights.

TABLE OF CONTENTS

INFLITE: AN INTELLIGENT INSTRUMENT FLIGHT TRAINER WITH COMPUTER-GENERATED SPEECH

INTRODUCTION

Researchers at the Air Force Human Resources Laboratory (AFHRL) are applying a taxonomy of learning skills (Kyllonen & Shute, 1987) to the pedagogical issues surrounding Intelligent Tutoring System (ITS) design and development. The taxonomy provides us with a means of categorizing target domains and consequently specifying the appropriate training approaches for particular ITSs. Further, it highlights classes of domains for which ITSs are appropriate but have not yet been developed. This paper focuses on an application in an area that has only recently been explored: a class of tasks which we refer to as high-performance tasks (Regian & Shute, 1988). In high-performance tasks, there is a greater requirement for highly speeded, reliable, and automatic task performance than is found in the typical knowledge-rich ITS domains (e.g., medical diagnosis, electronic troubleshooting). This paper describes an Instrument Flight Trainer (INFLITE) developed at AFHRL's Training Systems Division to test the concept of using artificial intelligence to train high-performance tasks.

INFLITE is in no sense a serious attempt to develop an application-ready instrument flight training device. Rather, INFLITE is an experimental system designed for the purpose of evaluating a promising approach to training high-performance tasks. Our decision to use flight simulation as the prototype domain was guided by a desire to use an inherently interesting task to increase motivation in our experimental subjects.

TAXONOMY OF TASKS

A task taxonomy provides a structure for classifying and categorizing various tasks according to a fixed set of dimensions, categorizing the domains and consequently specifying the appropriate training approaches for these tasks. An obvious benefit of having a viable taxonomy of tasks would be for designing, implementing and evaluating computerized ITSs. A number of ITSs have already been developed (Yazdani, 1986), and the potential for generalizing and synthesizing results across the different systems is being seen as increasingly critical to the further development of this technology (Soloway & Littman, 1986). Many researchers developing these powerful, innovative instructional systems have neither the interest nor the expertise for systematically evaluating those systems, nor is there an accepted framework within which to classify such evaluations. There have been only a few small-scale evaluation studies of global outcomes (e.g., Anderson, Boyle, & Reiser, 1985). By referencing an appropriate taxonomy, system developers could categorize the skills being developed and evaluators could determine the degree of success achieved. ITS approaches that are successful for a category of skills (or tasks) could then be applied to other skills or tasks that fall in the same category. In this way, a taxonomy could provide a useful framework within which to compare and evaluate tutors as to their relative effectiveness in teaching the categorized subject matter, and ultimately result in a guidebook of training approaches organized by task type.

Our proposed taxonomy (Kyllonen & Shute, 1987) simultaneously characterizes tasks, students, and learning environments along four dimensions: **Knowledge type** (i.e., declarative knowledge to procedural knowledge to automatic skill); **Instructional environment** (i.e., rote

learning; learning from instruction; learning by deduction; learning by drill and practice; or inductive learning by analogy, examples and discovery); **Domain or subject matter** (i.e., a two-dimensional space involving quality versus speed in decision making as one dimension and quantitative versus non-quantitative subject matter as the other); and **Learning styles** (e.g, holistic versus serial processing, active/impulsive versus passive/reflective orientation, systematic versus haphazard approach, spatial versus verbal representation, and so on). The major utility will be in codifying the interactions among dimensions, for some subject matters lend themselves more readily to certain instructional approaches.

The taxonomy already provides a principled means of categorizing tasks. Our next step is to begin cataloging training approaches that have been demonstrated to be effective for the various task categories. In light of the huge body of literature that is available to the training community, it is far more useful to organize the results according to task type rather than by specific tasks. One such task type we refer to as "high-performance tasks."

TUTORS FOR HIGH-PERFORMANCE TASKS

Traditional ITSs have focused on knowledge-rich domains where the majority of the knowledge engineering effort is centered on the skills and knowledge required to perform the target task. For example, in developing ITSs for electronic troubleshooting or medical diagnosis, one spends a great deal of time developing a performance model of how experts go about diagnosis, and determining what knowledge they employ to support this enterprise. Relatively little effort is typically expended in modeling instructional expertise.

ITS developers have only recently addressed the possibility of applying artificial intelligence to the training of high-performance tasks (e.g., Ritter & Feurzeig, 1988). Examples of such tasks include air intercept control, air traffic control, mission control console operation, and remote vehicle operation. In these domains, the target task may be relatively simple in terms of underlying knowledge but skilled performance (in terms of speed, accuracy, reliability, or automaticity) is essential. Most of the knowledge engineering in this kind of domain would be in modeling expert instructional strategies. Instructors of high-performance tasks are constantly assessing the performance of their students and tailoring instruction on an individual basis (such as focusing now on speed, later on accuracy; or focusing on task components that seem weak in terms of some criterion). Further, the literature of cognitive psychology (and the training literature in general) is replete with findings on training of high-performance skills. These studies involve proceduralization and compilation (Anderson, 1983), skill acquisition (Newell & Rosenbloom, 1981), practice effects and the development of automaticity (Shiffrin & Schneider, 1977), resource load (Wickens, Sandry, & Vidulich, 1983), and so on.

At AFHRL, we are currently conducting two projects to investigate knowledge engineering techniques and ITS architectures for high-performance tasks, and to develop prototype tutors in high-performance tasks. The two tasks are the National Aeronautics and Space Administration (NASA) mission control propulsion console operations and instrument-based aircraft piloting. This work will greatly extend the range of domains for which ITSs can be applied. In addition to providing principles for development of tutors in high-performance tasks, we will apply these principles to intelligent tutoring of high-performance task components within knowledge-rich domains. The following section describes one of these prototype tutors, the Instrument Flight Trainer (INFLITE).

INFLITE

The INFLITE prototype system trains students to land a simulated aircraft (F-16) using instruments only. INFLITE was designed to run on an AT-class microcomputer. It is written in the C programming language and uses the CLIPS[1] expert system shell. The system was designed primarily to operate with a joystick, but also supports a keypad interface. For optimal utility, INFLITE requires a peripheral voice synthesis device capable of converting an ASCII character stream into articulated speech. Such devices are commonly available and quite inexpensive. The hardware platform allows the widespread and inexpensive use of the program, which satisfies our goal of practicality in ITS delivery platforms, but it presented challenges during software development. The program presents an accurate real-time simulation of the interactions among essential flight instruments, balancing processor time among the different functional units in order to avoid disruption of the display.

The display interface to INFLITE consists primarily of the Head-Up Display (HUD) which presents speed, altitude, heading, flight path, centerline indicator, and Instrument Landing System (ILS) beam indicators. In addition to joystick (or keypad) control, the student may toggle-on screen panel lights which depict landing gear, afterburner, wheel brake, and air brake status.

In the initial familiarization session, the student is given a guided tour of the display interface by an articulate coach. As the coach describes each component of the interface, that component is highlighted on the screen. Next, the student is given a series of practice exercises to engender familiarity with methods of controlling the simulation. When the coach is satisfied that the student is sufficiently acclimated, the student is allowed to begin training trials.

At the beginning of each training trial, the simulation commences with the aircraft in flight, under normal flight conditions and randomly positioned with respect to the target airstrip. The goal of each training trial is to successfully land the aircraft. To do this, the student must follow heading information from flight control, use the joystick to turn the aircraft to successive temporary headings, locate the ILS beam, and follow this beam down to the airstrip.

During each training trial, an intelligent coach monitors the student (e.g., airspeed, heading, deviation from ILS beam) and provides guidance in the same manner as an instructor pilot might guide a student pilot. This guidance is presented verbally, using a voice synthesis system to simulate human speech. The student also receives instructions from a ground-based flight controller, again using synthesized speech but with a different voice than that of the coach. During early training trials, the coach may choose to freeze the simulation to give guidance. During all trials, student performance information is recorded for later use in pre-briefing, interactive comments, and post-briefing by the flight coach. During the post-flight debriefs, the coach reviews the student's performance in comparison to performance on earlier flights and highlights specific areas to be worked on in future flights. During the training trials, the coach provides guidance and feedback to the student based on real-time observations, and anticipates problems based on the student's performance history. During the preflight briefs, the coach reminds the student of problem areas identified in earlier flights.

The intelligent tutoring is handled by the expert, coach, and student modeling modules built with the CLIPS expert system.[2] The expert outlines suggested pilot actions and judges

flight conditions. For example, the expert may suggest a change to a heading of 90 degrees to move the aircraft toward the ILS beam center. If the student does indeed choose to move toward the beam, the expert will note the heading chosen and the effect on the alignment of the craft with the ILS beam. The expert reports to the coach any motion toward or away from the landing goal and its subgoals. The student never hears directly from the expert. Instead, the expert provides information to the coach, who decides how best to interact with the student from an instructional perspective.

The coach monitors the student actions, looking for common errors and reporting deviations from the suggested actions with regard to the student modeling module. The coach may also choose to intervene with the student (unless the simulator is in a special no-feedback mode which is used for student evaluation). This intervention may be as simple as a warning, "Wes, you've drifted off course again." Alternatively, the prompt may involve the freezing of the display, with a lengthy description of the problem. The display is frozen only in early training trials, such as the first time the student encounters the problem of being aligned with the beam according to the ILS scales but moving away from the beam due to an incorrect heading.

The student modeling module records the student performance profile, which includes: common errors (mastered, unmastered, and presented but unlearned), deviations from the expert-suggested actions, short descriptions of the starting flight conditions and the history of the flight, and any coach conclusions.

SUMMARY

INFLITE is a prototypical example of a class of intelligent microprocessor-based training simulators which might fill the gap between classroom instruction and expensive simulator training. Such a class of simulators would be useful during initial training and for refresher training. INFLITE is a high-cognitive-fidelity/low-physical-fidelity simulator targeted to teach key cognitive skills required for high-performance tasks after declarative instruction and prior to high-physical-fidelity simulation instruction.

INFLITE will be used as an experimental testbed for purposes of evaluating the relative training effectiveness of various approaches to automated training of high-performance skills. For example, variations on the system are being developed with additional flight condition and effect simulations, increased student analysis, and increased student-coach interaction initiated by the student pilot.

REFERENCES

Anderson, J.R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University.

Anderson, J.R., Boyle, F., & Reiser, B. (1985). Intelligent tutoring systems. *Science,* **228**, 465-462.

Kyllonen, P.C., & Shute, V.J. (1987). A taxonomy of learning skills. In P. Ackerman, R. Sternberg, & R. Glaser (Eds.), *Learning and individual differences*. San Francisco: Freeman.

Newell, A., & Rosenbloom, P. (1981). Mechanisms of skill acquisition and the law of practice. In J.R. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, NJ: Erlbaum Associates.

Regian, J.W., & Shute, V.J. (1988). Artificial intelligence in training: The evolution of intelligent tutoring systems. *Proceedings of the Conference on Technology and Training in Education.* Biloxi, MI.

Ritter, F., & Feurzeig, W. (1988). Teaching real-time tactical thinking. In J. Psotka, L. Massey, & S. Mutter (Eds.), *Intelligent tutoring systems: Lessons learned*. Hillsdale, NJ: LEA.

Shiffrin, R.M., & Schneider, W. (1977). Controlled and automatic human information processing: II. Perceptual learning, automatic attending, and a general theory. *Psychological Review*, **84**, 127-190.

Soloway, E., & Littman, D. (1986). Evaluating ITSs: The cognitive science perspective. *Proceedings of the Research Planning Forum for Intelligent Tutoring Systems*. San Antonio, TX.

Wickens, C.D., Sandry, D., & Vidulich, M. (1983). Compatibility and resource competition between modalities of input, central processing, and output: Testing a model of complex task performance. *Human Factors*, **25**, 227-248.

Yazdani, M. (1986). Intelligent tutoring systems survey. *Artificial Intelligence Review*, **1**, 43-52.

NOTES

1. The C Language Integrated Production System (CLIPS) was developed by the Artificial Intelligence Section (AIS) of the Mission Planning and Analysis Division (MPAD) at NASA/JSC. INFLITE uses version 4.2, which was developed under joint funding from NASA and the US Air Force.

2. For interested programmers, the following is an example of a CLIPS rule definition in INFLITE. The keyword of the definition is "defrule" for "define rule." The keyword is followed by the name of the rule, and then the verbal description of the rule. This rule retracts the suggestion that an action be performed to increase the reading on a gauge. The student could have been flying too low relative to his distance to the airport, and may have made the correction himself. The "?request" line triggers the matching of this rule. The subsequent lines (up through the " = > ") grab the gauge reading and test for the suggested correction. If the correction has been made, the right-hand side of the rule is executed, the suggestion is retracted, and a message is printed on the screen.

```
( defrule EndLowValueCorrection
"Retract a suggested increase in a gauge reading"
?request <- ( correctionPerformed ?gauge positive ?magnitude )
?gaugestatus <- ( guagecondition ?gauge  tooLow  ?minValue )
( ?gauge ?curValue )
( test ( < = ?minValue ?curValue))
= >
(fprintout t crlf "LOW Speed correction followed..." crlf )
( retract ?gaugestatus )
( retract ?request )
```